DITTO: Diffusion Inference-Time T-Optimization for Music Generation

Zachary Novack [1 2 *] **Julian McAuley** [1] **Taylor Berg-Kirkpatrick** [1] **Nicholas J. Bryan** [2]

Abstract

We propose **D**iffusion **I**nference-**T**ime $\boldsymbol{T}$-**O**ptimization (**DITTO**), a general-purpose framework for controlling pre-trained text-to-music diffusion models at inference-time via optimizing initial noise latents. Our method can be used to optimize through any differentiable feature matching loss to achieve a target (stylized) output and leverages gradient checkpointing for memory efficiency. We demonstrate a surprisingly wide-range of applications for music generation including inpainting, outpainting, and looping as well as intensity, melody, and musical structure control – all without ever fine-tuning the underlying model. When we compare our approach against related training, guidance, and optimization-based methods, we find DITTO achieves state-of-the-art performance on nearly all tasks, including outperforming comparable approaches on controllability, audio quality, and computational efficiency, thus opening the door for high-quality, flexible, training-free control of diffusion models. Sound examples can be found at https://ditto-music.github.io/web/.

1. Introduction

Large-scale diffusion models (Ho et al., 2020) have emerged as a leading paradigm for generative media, with strong results in diverse modalities such as text-to-image (TTI) generation (Rombach et al., 2022; Karras et al., 2022; Chen, 2023), video generation (Ho et al., 2022; Gupta et al., 2023), and 3D object generation (Watson et al., 2022; Poole et al., 2022). Recently, there has been growing work in applying image-domain methods to audio by treating the frequency-domain spectrograms of audio as images, producing promising results in general text-to-audio (TTA) generation (Liu et al., 2023a;b; Huang et al., 2023b) and text-to-music (TTM) generation (Hawthorne et al., 2022; Forsgren & Martiros, 2022; Chen et al., 2023; Huang et al., 2023a; Schneider et al., 2023). These methods operate via pixel or latent diffusion (Rombach et al., 2022) over spectrograms with genre, mood, and/or keywords control articulated via text prompts.

However, these text-conditioned approaches typically only offer high-level control (e.g. style), motivating further work. Current attempts to add more precise control (e.g. time-varying conditions) for TTM diffusion models are promising yet present their own tradeoffs. Finetuning-based methods like ControlNet (Wu et al., 2023a; Saharia et al., 2022a; Zhang et al., 2023) require large-scale supervised training with labeled examples for each new control modality. Inference-time methods that guide the diffusion sampling process, on the other hand, struggle to achieve fine-grained expressivity due to relying on approximations of the model outputs during sampling (Levy et al., 2023; Yu et al., 2023).

In order to achieve an expressive control paradigm for TTM diffusion models that requires no supervised training and can accept arbitrary control signals at inference-time, we propose **DITTO**: **D**iffusion **I**nference-**T**ime $\boldsymbol{T}$-**O**ptimization. DITTO optimizes the initial noise latents $\boldsymbol{x}_T$ with respect to an *arbitrary* differentiable feature matching loss across any diffusion sampling process to control the model outputs, and ensures efficient memory use via gradient checkpointing (Chen et al., 2016). Despite generally being considered to encode little information (Song et al., 2020; Preechakul et al., 2022), we show the power and precision the initial noise latents have to control the diffusion process for a wide-variety of applications in music creation, enabling musically-salient feature control and high-quality audio editing. Compared to previous optimization-based works from outside the audio domain (Wallace et al., 2023a), DITTO achieves SOTA control while also being 2x as time and memory efficient. Overall, our contributions are:

- DITTO: a novel, training-free framework for controlling pre-trained TTM diffusion models that optimizes the initial noise latents to control the model outputs.
- We leverage gradient checkpointing for memory efficiency without compromising the sampling process.
- Application of DITTO to multiple fine-grained time-

[*] Work done during an internship at Adobe Research. [1] University of California – San Diego [2] Adobe Research. Correspondence to: Zachary Novack <znovack@ucsd.edu>, Nicholas J. Bryan <njb@ieee.org>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

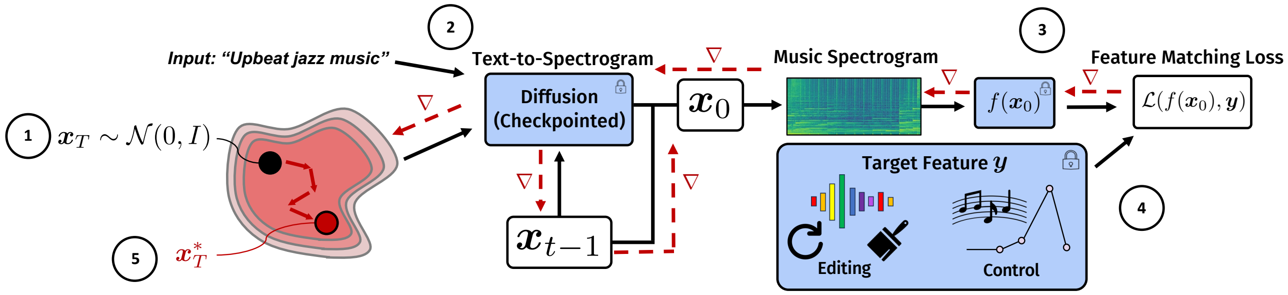

Figure 1. We propose **DITTO**, or **D**iffusion **I**nference-**T**ime ***T***-**O**ptimization, a general-purpose framework to control pre-trained diffusion models at inference-time. 1) We sample an initial noise latent x_T; 2) run diffusion sampling to generate a music spectrogram x_0; 3) extract features from the generated content; 4) input a target control signal; and 5) optimize the initial noise latent to fit any differentiable loss.

dependent tasks, including audio-domain inpainting, outpainting, melody control, intensity control, and the newly proposed looping and musical structure control.

- Evaluation showing our approach outperforms MultiDiffusion (Bar-Tal et al., 2023), FreeDoM (Yu et al., 2023), Guidance Gradients (Levy et al., 2023), Music ControlNet (Wu et al., 2023a), and the comparable optimization method DOODL (Wallace et al., 2023a), while being 2x faster and using half the memory.

2. Related Work

2.1. Music Generation Overview

Early works on generative music focused on *symbolic* generation (Dong et al., 2018; Chen et al., 2020; Dai et al., 2021). Recently, *audio*-domain music generation has become popular due to advances in language models (LMs) like MusicLM (Agostinelli et al., 2023) and diffusion models like AudioLDM (Liu et al., 2023a;b). LM-based approaches typically operate over discrete compressed audio tokens (Zeghidour et al., 2021; Kumar et al., 2023), generating audio either autoregressively (Borsos et al., 2023a; Agostinelli et al., 2023; Copet et al., 2023) or non-autoregressively (Garcia et al., 2023; Borsos et al., 2023b), and convert generated tokens back to audio directly. Diffusion-based approaches, on the other hand, typically operate by generating 2D frequency-domain representations of audio or *spectrograms* that are decoded into audio via a vocoder (Forsgren & Martiros, 2022; Liu et al., 2023a;b; Schneider et al., 2023).

2.2. Diffusion Models with Text Control

Text is currently the most popular control medium for diffusion models. Here, text captions are encoded into embeddings and injected into a generative model during training via cross attention, additive modulation, or similar as found in Stable Diffusion (Rombach et al., 2022) or Imagen (Saharia et al., 2022b). Despite its popularity, global caption-based text conditioning lacks fine-grained control (Zhang et al., 2023), motivating alternatives and the present work.

2.3. Alternative Train-time Control Methods

It is common to fine-tune existing text-conditioned diffusion models with additional inputs when adding advanced control. ControlNet-type models (Zhang et al., 2023; Zhao et al., 2023) use large sets of paired data to fine-tune TTI diffusion models by adding control adapters for specific predefined controls such as edge detection or pose estimation. To reduce training demands, a number of works fine-tune pretrained models on a small number of examples (Ruiz et al., 2023; Choi et al., 2023; Gal et al., 2022; Kawar et al., 2023). Others have explored using external reward models for fine-tuning, through direct fine-tuning (Clark et al., 2023; Prabhudesai et al., 2023) or reinforcement learning (Black et al., 2023). Such approaches, however, still require an expensive training process and the control mechanism cannot be modified after training. For music, only a ControlNet-style approach has been taken (Wu et al., 2023a). In contrast, DITTO requires no large-scale training and can accept any differentiable control at inference-time.

2.4. Inference-time Guidance-based Control

To avoid large-scale model fine-tuning, inference-time control methods have become increasingly popular. Early approaches include prompt-to-prompt image editing (Hertz et al., 2022) and MultiDiffusion (Bar-Tal et al., 2023), which enable localized object editing and in/outpainting by fusing multiple masked diffusion paths together. Such methods rely on control targets that can be localized to specific pixel regions of an image and are less applicable for audio spectrograms which have indirect pixel correspondences across frequency and multiple overlapping sources at once.

We also note the class of guidance-based methods (Dhariwal & Nichol, 2021; Chung et al., 2023; Levy et al., 2023; Yu et al., 2023), which introduce updates at each sampling step to steer the generation process via the gradient of a pretrained classifier $\nabla_{x_t}\mathcal{L}_\phi(x_t)$. These approaches generally require an approximation of model outputs during sampling, which are inaccurate at high noise levels and thus limit fine-grained expressivity. For music, guidance-based methods have only been explored in Levy et al. (2023). In contrast, DITTO calculates gradients with respect to the initial noise on the *real* model outputs through sampling, allowing accurate gradients to influence the entire generation process.

2.5. Inference-time Optimization-based Control

Recent work has shown optimization through diffusion sampling is possible if GPU memory is correctly managed. Direct optimization of diffusion latents (DOODL) (Wallace et al., 2023a) leverages the recently proposed EDICT sampler (Wallace et al., 2023b), which uses affine coupling layers (ACLs) (Dinh et al., 2014; 2016) to form a fully invertible sampling process, and backpropagates through EDICT to optimize initial noise latents for improving high-level features like CLIP guidance and aesthetic improvement in images. DOODL, in contrast to our approach, struggles on fine-grained control signals (Wallace et al., 2023a) and has multiple downsides due to its reliance on EDICT including 1) it is restricted to only invertible sampling algorithms; 2) it requires double the model evaluations for both forward and reverse sampling that increase latency and memory use; and 3) it can suffer from stability issues and reward hacking due to divergence between the ACL diffusion chains.

Karunratanakul et al. (2023) proposed backpropagating through sampling for human motion generation (i.e. short sequences of joint positions). This work leverages numerous domain-specific modifications to reduce memory usage, such as using a small (i.e. <18M parameters) transformer encoder-only architecture, very few sampling steps, long optimization time, and purely unconditional generation. Thus, this approach is not applicable to more standard generative tasks with higher memory demands like text-to-image/audio/music, while DITTO circumvents any restrictions on the model architecture or sampler.

3. Diffusion Inference-Time T-Optimization

3.1. Diffusion Background

Denoising diffusion probabilistic models (DDPMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020) or diffusion models are defined by a forward and reverse random Markov process. The forward process takes clean data and iteratively corrupts it with noise to train a neural network $\boldsymbol{\epsilon}_\theta$. The network $\boldsymbol{\epsilon}_\theta$ typically inputs (noisy) data $\boldsymbol{x}_t$, the diffusion step t, and (text) conditioning information $\boldsymbol{c}$. The reverse process takes random noise $\boldsymbol{x}_T \sim \mathcal{N}(0, I)$ and iteratively refines it with the learned network to generate new data $\boldsymbol{x}_0$ over T time steps (e.g., 1000) via the sampling process,

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\Big(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})\Big) + \sigma_t\boldsymbol{\epsilon}, \quad (1)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, $\alpha_0 := 1$, α_t and $\bar{\alpha}_t$ define the noise schedule, σ_t is the sampling standard deviation. To reduce sampling time, Denoising Diffusion Implicit Model (DDIM) sampling (Song et al., 2020) uses an alternative optimization objective that yields a faster sampling process (e.g., $20-50$ steps) that can be deterministic. Broadly, we can denote any sampling algorithm with the notation $\boldsymbol{x}_{t-1} = \texttt{Sampler}(\boldsymbol{\epsilon}_\theta, \boldsymbol{x}_t, t, \boldsymbol{c})$.

To improve text conditioning, classifier-free guidance (CFG) can be used to blend conditional and unconditional generation outputs (Ho & Salimans, 2021). When training with CFG, conditioning is randomly set to a null value a fraction of the time. During inference, the diffusion model output $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})$ is linearly combined with $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}_\emptyset)$ using the CFG scale w, where $\boldsymbol{c}_\emptyset$ are null embeddings. Note, CFG during inference doubles the forward passes of $\boldsymbol{\epsilon}_\theta$. For a diffusion model review, see Appendix A.

3.2. Problem Formulation

Instead of trying to control diffusion models by using expensive supervised training or inexact inference-time guidance-based methods, we alternatively formulate the control task as an *optimization* problem. Notably, we can denote the output of the model after running the sampler for a total of T sampling steps as $\boldsymbol{x}_0 = \texttt{Sampler}_T(\boldsymbol{\epsilon}_\theta, \boldsymbol{x}_T, \boldsymbol{c})$, showing that the final output is a function of the *initial* noise latents $\boldsymbol{x}_T \sim \mathcal{N}(0, I)$.

While $\boldsymbol{x}_T$ is normally just considered to be a random seed, we can instead treat the initial noise latents as a free parameter to be optimized at inference-time. In particular, we define a target feature extractor $f(\cdot)$, which only needs to be differentiable, and some corresponding loss function $\mathcal{L}$ to measure how well the model output's particular feature matches a target control $\boldsymbol{y}$. With this, we can then directly optimize $\boldsymbol{x}_T$ *through* the sampling process such that the model output $\boldsymbol{x}_0$ follows the target control. Formally,

$$\boldsymbol{x}_T^* = \arg\min_{\boldsymbol{x}_T} \mathcal{L}\left(f(\boldsymbol{x}_0), \boldsymbol{y}\right) \quad (2)$$

$$\boldsymbol{x}_0 = \texttt{Sampler}_T(\boldsymbol{\epsilon}_\theta, \boldsymbol{x}_T, \boldsymbol{c}) \quad (3)$$

By framing the control task as an arbitrary feature-matching optimization on the initial noise latents, we are able to incorporate a diverse range of control tasks by constructing $f(\cdot)$ and $\mathcal{L}$ accordingly, such as letting f extract the intensity curve of the music and $\mathcal{L}$ being the squared ℓ_2 distance to some target intensity (see Sec. 4 for more details). This procedure requires no training (as only $\boldsymbol{x}_T$ is optimized rather than model weights) and uses *exact* control gradients (as $f(\cdot)$ is only called on the real output).

Algorithm 1 Diffusion Inference-Time T-Optimization (DITTO)

```
input : $\epsilon_\theta$, Sampler, sampling steps $T$, feature extractor $f$, loss $\mathcal{L}$, target feature $\boldsymbol{y}$, starting latent $\boldsymbol{x}_T$, text conditioning $\boldsymbol{c}$, optimization steps $K$, optimizer $g$.
// Run optimization
for $i = 1$ to $K$ do
    // Initialize noise latents
    $\boldsymbol{x}_t \leftarrow \boldsymbol{x}_T$
    // Diffusion sampling w/grad checkpointing per step
    for $t = T$ to 1 do
        $\boldsymbol{x}_{t-1} = \texttt{Checkpoint}(\texttt{Sampler}, \boldsymbol{\epsilon}_\theta, \boldsymbol{x}_t, t, \boldsymbol{c})$
    end for
    // Extract features from generated output
    $\hat{\boldsymbol{y}} = f(\boldsymbol{x}_0)$
    // Compute the loss and backprop
    $\boldsymbol{x}_T \leftarrow \boldsymbol{x}_T - g(\nabla_{\boldsymbol{x}_T} \mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y}))$
end for
output : $\boldsymbol{x}_0$
```

Solving (2) using backpropagation, however, is typically intractable due to extreme memory requirements. Namely, the diffusion sampling process is recursive by design and standard automatic differentiation packages customarily require storing all intermediate results for each of T recurrent calls to $\boldsymbol{\epsilon}_\theta$ within the sampler ($2T$ sets of activations per step when CFG is used). Thus, even 2-3 sampling steps can cause memory errors with standard U-Net diffusion architectures.

3.3. Diffusion with Gradient Checkpointing

To circumvent large memory use during optimization, we use gradient checkpointing (Chen et al., 2016). The core idea is to discard intermediate activation values stored during the forward pass of backpropagation that inflict high memory use and recalculate them during the backward pass when needed from cached inputs. We use gradient checkpointing on each model call during sampling, as the memory required to store the intermediate noisy diffusion tensors and conditioning information is minute compared to the intermediate activations of a typical diffusion model (e.g., cross-attention activation maps within a large UNet). Our memory cost to optimize (2) with sampler-step checkpointing is 1) the memory needed to run backpropagation on one diffusion model call $\boldsymbol{\epsilon}_\theta$ plus 2) the cost to store the T intermediate noisy diffusion tensors $\boldsymbol{x}_t \forall t = 0, ..., T$ and conditioning $\boldsymbol{c}$. While we pay for the memory reduction with an additional forward pass per time step (as shown in Fig. 2), this straightforward trick allows DITTO to maintain efficiency without changing any part of the sampling algorithm.

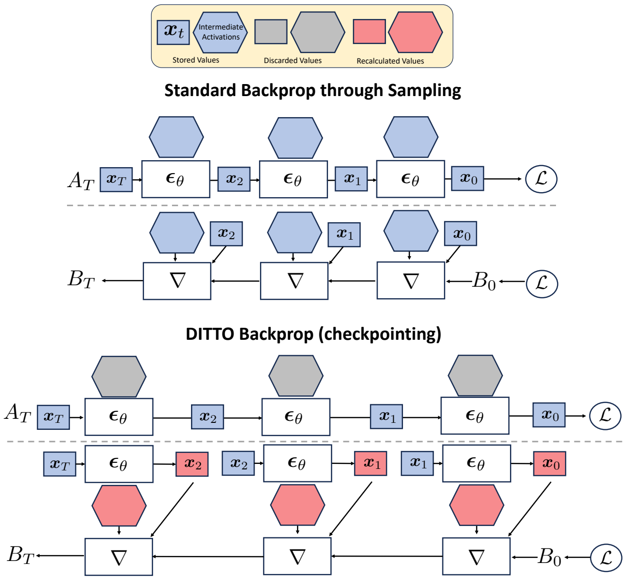

Figure 2. Different memory setups for backpropagation through sampling. Normally, all intermediate activations are stored in memory, which is intractable for modern diffusion models. In DITTO, gradient checkpointing allows us to achieve efficient memory usage with only 2x the number of model calls to preserve fast runtime.

In contrast to our approach, DOODL explored gradient checkpointing via the MemCNN library (Leemput et al., 2019). However, their use of the EDICT sampler *doubles* the memory and runtime cost compared to our method (see Appendix B) and adds instability to the sampling process due EDICT's dual-chain sampling (see Section 6.4).

3.4. Complete Algorithm

Psuedo-code for our DITTO algorithm is shown in Algorithm 1. We define `Checkpoint` to be a gradient checkpointing function that 1) inputs and stores a callable differentiable network (i.e., the sampler) and any input arguments to the network, 2) overrides the default activation caching behavior of the network to turn off activation caching during the forward pass of backpropagation and 3) recomputes activations when needed in the backward pass. Note that in practice, we typically use a small subsequence of sampling steps (e.g. 20) spanning from $\boldsymbol{x}_T$ to $\boldsymbol{x}_0$.

4. Applications and Control Frameworks

We apply our flexible paradigm to a range of applications[1] by parameterizing each control framework (i.e. f and $\mathcal{L}$) to directly target musically-salient features, allowing for outpainting, inpainting, looping, intensity control, melody control, and musical structure control, where musical structure

[1] We leave the rhythm control from Wu et al. (2023a) for future work, as their RNN beat detector would trigger an exceedingly long backpropagation graph when using DITTO.

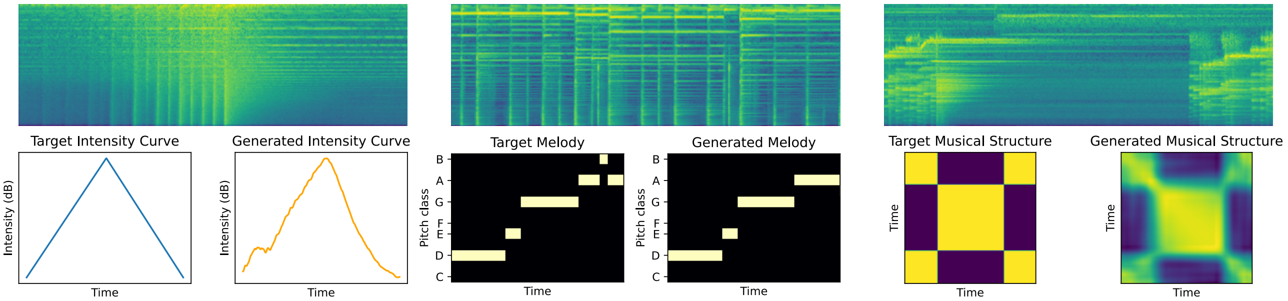

Figure 3. Examples of DITTO's use for creative control, including intensity (left), melody (middle), and structure (right), with target controls and final features displayed below each spectrogram. All results are achieved without additional training or fine-tuning.

and looping have been unexplored for TTM diffusion models. These constitute both reference-based (i.e. using existing audio) and reference-free (generation from scratch, as shown in Fig. 3) control operations. Our goal here is to display the expressive controllability that initial noise latents have over the diffusion process.

Outpainting – Outpainting is the task of extending the length of existing audio and is critical for audio editing as well as generating long-duration music content using diffusion models. Past outpainting methods include MultiDiffusion (Bar-Tal et al., 2023) and Guidance Gradients (Levy et al., 2023) which struggle to maintain long-form coherence and local smoothing. We perform outpainting by 1) taking an existing reference audio signal $\boldsymbol{x}_{\text{ref}}$; 2) defining an overlap region o in seconds at the end of the reference; 3) using DITTO to create new content that matches the overlap region at the *beginning* of the new generation; and 4) stitching the reference and newly generated content together. More formally, we define $\mathbf{M}_{\text{ref}}$ and $\mathbf{M}_{\text{gen}}$ as binary masks that specify the location of the overlap region in the reference and generated content respectively, $f(\boldsymbol{x}_0) := \mathbf{M}_{\text{gen}} \odot \boldsymbol{x}_0$, $\boldsymbol{y} = \mathbf{M}_{\text{ref}} \odot \boldsymbol{x}_{\text{ref}}$, and $\mathcal{L} \propto ||f(\boldsymbol{x}_0) - \boldsymbol{y}||_2^2$.

Inpainting – Inpainting is the task of replacing an interior region of real or previously generated content and is essential for audio editing and music remixing. Past work on inpainting has been explored in the image- and audio-domain to variable success (Chung et al., 2023; Levy et al., 2023). We use DITTO to perform inpainting similar to outpainting, with the only modification being $\mathbf{M}_{\text{ref}} = \mathbf{M}_{\text{gen}}$ denote *two* overlap regions (on each side of the spectrogram) to use as context for inpainting the gap in between.

Looping – Looping is the task of generating content that repeats in a circular pattern, creating repeatable music fragments to form the basis of a larger composition. For looping, we use DITTO similar to outpainting, but when we define $\mathbf{M}_{\text{ref}}$ and $\mathbf{M}_{\text{gen}}$, we specify two overlapping edge regions of the output (similar to inpainting) but corresponding to *opposite* sides of the outputs (similar to outpainting), such that the extended region seamlessly transitions back to the beginning of the reference clip. To our knowledge, we are the first to imbue TTM diffusion models with looping control.

Intensity Control – Musical intensity control is the task of adjusting the dynamic contrast of generated music across time. We follow the intensity control protocol from Music ControlNet (see Wu et al. (2023a) for more details), which employs a training-time method to generate music that follows a smoothed, decibel (dB) volume curve. In our case, we use DITTO in a similar fashion, albeit without the need for large-scale fine-tuning, by setting $f(\boldsymbol{x}_0) := \boldsymbol{w} * 20\log_{10}(\texttt{RMS}(\mathbf{V}(\boldsymbol{x}_0)))$, where $\boldsymbol{w}$ are the smoothing coefficients used in Music ControlNet, $*$ is a convolution operator, $\texttt{RMS}$ is the Root Mean Squared energy of the audio, $\boldsymbol{y}$ is a given dB-scale target curve, $\mathcal{L} \propto ||f(\boldsymbol{x}_0) - \boldsymbol{y}||_2^2$, and $\mathbf{V}$ is our vocoder (Lee et al., 2022; Zhu et al., 2024) that translates spectrograms to the audio domain. Here, we backpropagate through our vocoder as well. Notably, under this parameterization intensity control does not only control the loudness of the generated audio but also the harmonic and rhythmic density of the music (which is correlated with RMS energy).

Melody Control – Musical melody control is the task of controlling prominent musical tones over time and allows creators to generate accompaniment music to existing melodies. Following recent work (Copet et al., 2023; Wu et al., 2023a), the approx. melody of a recording can be extracted by computing the smoothed energy level of the 12-pitch classes over time via a highpass chromagram function $\mathbf{C}(\cdot)$ (Müller, 2015). Given this, we use DITTO with $f(\boldsymbol{x}_0) = \log(\mathbf{C}(\mathbf{V}(\boldsymbol{x}_0)))$, a target melody $\boldsymbol{y} \in \{1, \dots, 12\}^{N \times 1}$, the spectrogram length N, and $\mathcal{L} = \text{NLLLoss}(f(\boldsymbol{x}_0), \boldsymbol{y})$ or the negative log likelihood loss. See Wu et al. (2023a) for further implementation details.

Musical Structure Control – We define musical structure control as the task of controlling the high-level musical form of generated music over time. To model musical form, we follow musical structure analysis work (McFee & Ellis, 2014) that, in the simplest case, measures structure via computing a self-similarity (SS) matrix of local timbre features where timbre is "everything about a sound which is neither loudness nor pitch" (Erickson, 1975). Thus, we use DITTO for musical structure control by setting $\mathbf{y}$ to be a known, target SS matrix, $f(\boldsymbol{x}_0) = \mathbf{T}(\boldsymbol{x}_0)\mathbf{T}(\boldsymbol{x}_0)^\top$, $\mathbf{T}(\cdot)$ to be a timbre extraction function, and $\mathcal{L} \propto ||f(\boldsymbol{x}_0) - \boldsymbol{y}||_2^2$. Specifically, we use the Mel-Frequency Cepstrum Coefficients (MFCCs) (McFee et al., 2010), omitting the first coefficient and normalized across the time axis, as the timbre extraction function, and then smooth the SS matrix via a 2D Savitzky-Golay filter in order to not penalize slight variations in intra-phrase similarity. Such target SS matrices can take the form of an "ABBA" pattern (as shown in Fig. 3) for instance. To our knowledge, we are the first to imbue TTM diffusion models with structure control.

Other Applications – Besides the applications described above, DITTO can be used for numerous new extensions previously unexplored in TTM generation which we describe in the Appendix, such as correlation-based intensity control (C), real-audio inversion (D), reference-free looping (E), musical structure transfer (F), other sampling methods (G), multi-feature optimization (H), and reusing optimized latents for fast inference (I).

5. Experimental Design

5.1. DITTO Setup

We use Adam (Kingma & Ba, 2014) as our optimizer for DITTO, with a learning rate of 5×10^{-3} (as higher leads to stability issues). We use DDIM (Song et al., 2020) sampling with 20 steps and dynamic thresholding (Saharia et al., 2022b) for all experiments. No optimizer hyperparameters were changed across application besides the max number of optimization steps, which were doubled from 70 to 150 for the melody and structure tasks.

5.2. Datasets

We train our models on a dataset of ≈1800 hours of licensed instrumental music with genre, mood, and tempo tags. Our dataset does not have free-form text description, so we use class-conditional text control of global musical style, as done in JukeBox (Dhariwal et al., 2020). For melody control references, we synthesize recordings from a 380-sample public-domain subset of the **Wikifonia Lead-Sheet Dataset** (Simonetta et al., 2018). Like in Wu et al. (2023a), we construct a small set of handcrafted intensity curves and musical structure matrices (e.g. a smooth crescendo and "ABA" form) for intensity and structure control (see Appendix H for more examples). For evaluation only, we also use the **MusicCaps Dataset** (Agostinelli et al., 2023) with around 5K 10-second clips with text descriptions.

5.3. Evaluation Metrics

We use Frechet Audio Distance (FAD) with the CLAP music (Wu et al., 2023b) backbone (as the default VGGish backbone is documented to poorly correlate with human perception (Gui et al., 2023)), which measures the distance between the distribution of embeddings from a set of baseline recordings and that from generated recordings (Kilgour et al., 2018). FAD metrics are calculated using MusicCaps as the reference distribution against 2.5K model generations for all experiments. For reference-free targets, we also use the CLAP score (Wu et al., 2023b), which measures the overall alignment between the text caption and the output audio; note that as our model is only *tag*-conditioned, we convert each tag set into a caption using the template *"A [genre] [mood] song at [BPM] beats per minute"*. Additionally, for the intensity and musical structure control, we report the average loss $\mathcal{L}$ across the generated outputs (i.e. the final feature matching distance), and report overall accuracy for melody control, since it is framed as a classification task.

5.4. Baselines

We benchmark against a wide-range of methods including:

- Naïve Masking: Here, after a DDIM-step we apply the update $\boldsymbol{x}_{t-1} = \mathbf{M}_{\text{ref}} \odot \mathcal{N}(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_{\text{ref}}, (1-\bar{\alpha}_t)\boldsymbol{I}) + \mathbf{M}_{\text{gen}} \odot \boldsymbol{x}_{t-1}$ (i.e. setting the overlap region directly to the reference image at the appropriate noise level).
- MultiDiffusion (Bar-Tal et al., 2023): This case is similar to the naïve approach, but instead *averages* the noisy outputs in the overlapping region instead of using a hard mask. We can additionally stop this averaging operation at certain points of the sampling process (such as half way through) and let the model sample without guiding the process; we denote the former approach as MD and the latter as MD-50 for brevity.
- FreeDoM (Yu et al., 2023): FreeDoM is a guidance-based method, where we perform an additional update during sampling $\boldsymbol{x}_t = \boldsymbol{x}_t - \eta_t \nabla_{\boldsymbol{x}_t} \mathcal{L}(f(\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)), \boldsymbol{y})$, where $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)$ denotes the first term in Eq. 12. η_t is a time-dependent learning rate that is a function of the overall gradient norm.
- Guidance Gradients (GG) (Levy et al., 2023): GG takes the update equation from FreeDoM and makes two small modifications. Namely, η_t is fixed throughout sampling, and GG includes an additional data consistency step when the feature extractor $f(\cdot)$ is fully linear.
- Music ControlNet (Wu et al., 2023a): Music ControlNet is a training-based approach that shares the same underlying base model as our work but additionally fine-tunes adaptor modules during large scale training to the control signal $\boldsymbol{y}$ as conditioning.

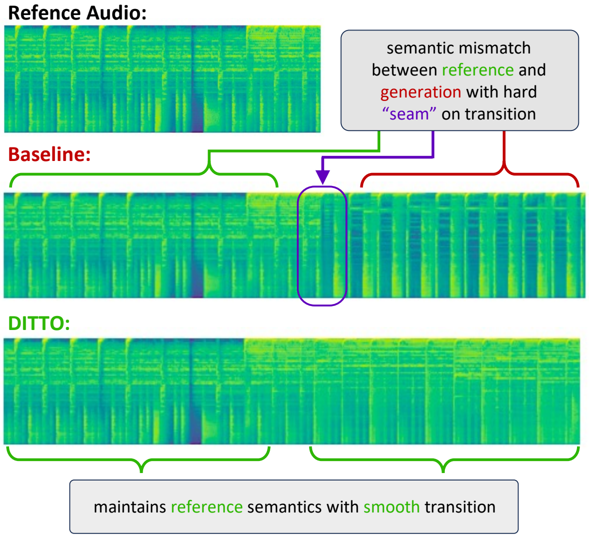
Refence Audio:
semantic mismatch between reference and generation with hard "seam" on transition
Baseline:
DITTO:
maintains reference semantics with smooth transition

Figure 4. Failure cases of baseline outpainting methods. Baseline methods tend to create audible "seams" in the audio between overlap and non-overlap regions of the generated output, leading to unnatural jumps in semantic content. DITTO avoids this issue and provides seamless outpainting throughout the full generation.

Table 1. Outpainting and looping FAD (↓) results for DITTO against baseline pixel, guidance, and optimization-based methods.

Method	$o = 1$	$o = 2$	$o = 3$	Looping
DOODL	0.719	0.707	0.700	0.750
Naive	0.722	0.716	0.712	0.753
MD	0.733	0.716	0.710	0.749
MD-50	0.718	0.714	0.705	0.752
GG	0.754	0.738	0.719	0.774
FreeDoM	0.726	0.723	0.715	0.758
DITTO (ours)	**0.716**	**0.703**	**0.698**	**0.746**

Table 2. Inpainting FAD (↓) results for DITTO against baseline pixel, guidance, and optimization-based methods.

Method	gap = 2	gap = 3	gap = 4
DOODL	0.688	0.693	0.696
Naive	0.697	0.705	0.707
MD	0.690	0.694	0.701
MD-50	0.701	0.708	0.711
GG	0.700	0.709	0.717
FreeDoM	0.704	0.709	0.719
DITTO (ours)	**0.686**	**0.688**	**0.690**

- DOODL (Wallace et al., 2023a): DOODL[2] is an optimization-based approach that uses the EDICT (Wallace et al., 2023b) sampler and multiple ad-hoc changes to the optimization process such as injecting noise and renormalizing $\boldsymbol{x}_T$. We use the same learning rate as DITTO due to similar stability issues.

We compare with Naïve Masking, MultiDiffusion, and Guidance Gradients for inpainting, outpainting, and looping experiments since they all have linear feature matching objective, Music ControlNet for the melody and intensity experiments, and FreeDoM and DOODL for all experiments.

6. Results

6.1. Outpainting, Inpainting, and Looping Results

We show objective evaluation results for outpainting and looping in Table 1 and inpainting results in Table 2. Here we report FAD, as low loss over the overlap regions does not necessitate that the *overall* audio is cohesive. We find DITTO achieves the lowest FAD against all baselines across overlap sizes of 1 to 3 seconds and inpainting gaps of 2 to 4 seconds. DOODL performs next behind DITTO, and the inference-time guidance methods particularly struggle.

Qualitatively, we discover that all baselines (besides DOODL) tend to produce audible "seams" in the output music outside the overlap region as shown in Fig. 4, wherein the final outputs tend to purely match the overlap region (i.e. over optimizing for the feature matching target) and ignore the overall consistency between the overlap generation and the rest of the generation. By optimizing $\boldsymbol{x}_T$ for reconstruction over the overlap regions, DITTO effectively avoids such issues, as this process implicitly encourages the non-overlap generation sections to preserve semantic content seamlessly.

6.2. Intensity, Melody, and Structure Results

In Table 3, we show objective metrics for intensity, melody, and structure control. We seek to understand 1) how different methods impose the target control on the generative model via MSE or Accuracy 2) overall audio quality via FAD and 3) how such control effects the baseline text conditioning via CLAP. We find DITTO achieves SOTA intensity and melody control, beating that of Music ControlNet with *zero* supervised training. We further explore Music ControlNet's poor intensity control more in-depth in Appendix C. Additionally, we note FreeDoM slightly beats DITTO in structure control, but exhibits poor performance for intensity and especially melody control, showing the limits of guidance-based methods for complicated feature extractors.

A notable concern with optimization-based control is the chance of reward hacking (Skalse et al., 2022; Prabhudesai et al., 2023), where the control target is over-optimized leading to degradation in model quality and base behavior. We find that DOODL exhibits this reward hacking behavior consistently in addition to generally being worse at control than DITTO, sacrificing overall quality and significant text relevance in favor of matching the control target. DITTO, on

[2] https://github.com/salesforce/DOODL

Table 3. Intensity, melody, and structure control results. DITTO achieves SOTA intensity and melody control. Music ControlNet struggles on intensity control MSE. FreeDoM performs well on structure but struggles on more complex melody and intensity control.

Control	**Intensity**			**Melody**			**Structure**		
Metric	MSE (↓)	FAD (↓)	CLAP (↑)	Acc (↑)	FAD (↓)	CLAP (↑)	MSE (↓)	FAD (↓)	CLAP (↑)
Default TTM	40.843	0.707	0.373	10.527	0.707	0.373	0.309	0.707	0.373
ControlNet	38.411	**0.637**	0.308	81.353	**0.545**	**0.478**	–	–	–
FreeDoM	23.292	0.673	**0.482**	31.544	0.706	0.477	**0.018**	0.668	0.415
DOODL	4.785	0.695	0.342	81.592	0.715	0.336	0.074	0.653	0.387
DITTO (ours)	**4.758**	0.682	0.433	**82.625**	0.699	0.432	0.024	**0.632**	**0.418**

the other hand, is able to balance the target control without over-optimizing and maintain quality and text relevance.

In Fig. 3, we show qualitative intensity, melody, and structure control results. On the left, we show a generated spectrogram with a rising then falling intensity curve. In the middle, we show a generated spectrogram with an input target and generated melody visualization (chromagram). On the right, we show a generated spectrogram with target and generated self-similarity matrices with an ABBA structure pattern.

6.3. Subjective Listening Test

Given that audio quality is subjective, we performed a small scale listening test to measure the efficacy of DITTO against alternative methods. Specifically, we asked test participants to rate the audio quality for three different applications including Intensity, Outpainting, and Melody across several algorithms. We generated 10 random samples for each applications using the same text prompts and control for each method. We compare DITTO with FreeDoM and Music ControlNet for Intensity and Melody control, and with FreeDoM and MD-50 for outpainting. For each triplet of outputs for the given controls, participants were asked to rate the overall quality of the generated music for each output on a 0-100 scale. We recruited 15 participants for the listening study, thus totaling 150 scores per setting and control method.

In Table 4, we show the number of wins for DITTO and the average difference in rating scores between DITTO and each other method (where positive score difference denotes DITTO is higher). Notably, we find that DITTO is strongly preferred against FreeDoM on all tasks, Music ControlNet on Intensity, and MD-50 on Outpainting. On Melody control, we find practically no difference between DITTO and Music-ControlNet. This provides evidence that DITTO has superior or equal quality over SOTA controllable music generation methods.

6.4. Efficiency Comparison

Besides comparing DITTO with DOODL in terms of their generation quality and control, we seek to understand how they differ in terms of both practical efficiency and convergence speed, as slow per-iteration runtime could be offset by fast convergence, and how such behaviors change as the number of sampling steps increases. We focus on intensity control since it represents a middle ground between the simple linear painting methods and the more complex melody control. Besides MSE, FAD, and CLAP, we also report the mean steps to convergence (MS2C), i.e. the average number of optimization steps needed to reach an MSE below some threshold τ, the mean optimization speed (MOS), i.e. the average number of seconds per optimization step, and the mean allocated memory (MAM), measuring the average GPU memory (in GB) used during optimization by the diffusion model. See Appendix L for more details.

Table 4. Subjective listening test results. DITTO is strongly preferred to FreeDoM and Music ControlNet / MD-50 on outpainting and intensity tasks, and is roughly equivalent to Music ControlNet on melody control.

	Intensity	
Comparison Test	% Wins	Avg. Difference
DITTO vs. ControlNet	65	**15.90** (± 2.79)
DITTO vs. FreeDoM	71	**20.35** (±2.55)
	Outpainting	
Comparison Test	% Wins	Avg. Difference
DITTO vs. MD-50	77	**23.49** (± 2.57)
DITTO vs. FreeDoM	80	**26.17** (± 2.33)
	Melody	
Comparison Test	% Wins	Avg. Difference
DITTO vs. ControlNet	48	1.40 (± 2.28)
DITTO vs. FreeDoM	61	**9.55** (± 2.05)

In Table 5, we empirically confirm that DOODL is $\approx$ 2x slower than DITTO and takes up $\approx$ 2x more GPU memory, as DOODL uses the EDICT sampler which doubles the number of model calls during both the forward and checkpointed backwards pass and stores both chains of inputs in memory. Most saliently, we discover that DOODL displays practically identical convergence speed to DITTO, showing that DOODL's added complexity provides no benefit in speeding up optimization. We note that increasing the number of sampling steps tends to degrade control adherence, likely since the longer sampling chain makes backpropaga-

Table 5. Performance between DITTO and DOODL on intensity control. DITTO and DOODL reach convergence in a similar number of steps yet DOODL is ≈2x less efficient than DITTO.

Method Sampling Steps	DITTO 20	DOODL 20	DITTO 50	DOODL 50
MSE (↓)	**4.758**	4.785	**7.640**	8.894
FAD (↓)	**0.682**	0.695	0.661	**0.636**
CLAP (↑)	**0.433**	0.342	**0.398**	0.311
MS2C (↓)	**44.466**	49.203	**46.855**	47.834
MOS(↓)	**1.859**	4.177	**4.472**	10.036
MAM (↓)	**5.002**	8.274	**5.094**	8.311

tion more difficult. Interestingly, as sampling time increases the overall FAD improves significantly for DOODL, giving evidence that EDICT particularly struggles with few sampling steps, and thus DOODL cannot be sped up by using fewer steps without noticeable reward hacking.

We note that inference-time optimization-based techniques are slower than both guidance-based techniques and training-based techniques at inference time by design, as they functionally amortize the cost of the training-based methods (which require hundreds of GPU hours to fine-tune) at inference-time to offer more expressivity than the guidance-based methods (see Appendix L for more discussion). Given that the speed of DITTO is primarily tied to the number of sampling steps used to sample the model (as well as the need for gradient checkpointing), there are clear ways to accelerate DITTO using the growing line of work in fast diffusion samplers (Lu et al., 2022; Luo et al., 2023; Kim et al., 2023), which we leave for future work.

6.5. The Expressive Power of the Diffusion Latent Space

Typically, the initial latent $\boldsymbol{x}_T$ is ignored in diffusion models, as the diffusion latent space has previously been thought to encode little semantic meaning compared to GAN latent spaces (Song et al., 2020; Preechakul et al., 2022). DITTO's strong performance, however, presents the surprising fact that a wide-array of semantically meaningful fine-grained features can be manipulated purely through exploring the diffusion latent space without ever editing the pre-trained diffusion base model. We explore this idea further, and how our findings are theoretically tied to the encoding of low-frequency structure noted by Si et al. (2023) in Appendix J.

7. Conclusion

We propose DITTO: **D**iffusion **I**nference-**T**ime $\boldsymbol{T}$-**O**ptimization, a unified training-free framework for controlling pre-trained diffusion models to enable a wide-range of creative editing and control tasks for music generation. DITTO achieves SOTA editing ability and matches the controllability of fully training-based methods, outperforms the leading optimization-based approach while being 2x as time and memory efficient, and imposes no restrictions on the modeling architecture or sampling process. In future work, we hope to accelerate the optimization procedure to achieve real-time interaction and more expressive control.

Impact Statement

While generative multimedia models may open up new avenues for artistic creation, there is the concern of negatively impacting current working musicians and creators and their own livelihoods. We find that it is exceedingly important to build TTM systems that protect artists and their data. To mitigate harm, we train on licensed music and place our focus on improving controllability, allowing working artists to interface with TTM systems through more musically-aligned controls, instead of only relying on high-level textual prompts that may be too general for music professionals.

References

Agostinelli, A., Denk, T. I., Borsos, Z., Engel, J., Verzetti, M., Caillon, A., Huang, Q., Jansen, A., Roberts, A., Tagliasacchi, M., et al. MusicLM: Generating music from text. *arXiv:2301.11325*, 2023.

Bar-Tal, O., Yariv, L., Lipman, Y., and Dekel, T. MultiDiffusion: Fusing diffusion paths for controlled image generation. In *International Conference on Machine Learning (ICML)*, 2023.

Black, K., Janner, M., Du, Y., Kostrikov, I., and Levine, S. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

Borsos, Z., Marinier, R., Vincent, D., Kharitonov, E., Pietquin, O., Sharifi, M., Roblek, D., Teboul, O., et al. AudioLM: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 2023a.

Borsos, Z., Sharifi, M., Vincent, D., Kharitonov, E., Zeghidour, N., and Tagliasacchi, M. Soundstorm: Efficient parallel audio generation. *ArXiv*, abs/2305.09636, 2023b.

Chen, K., Wang, C.-i., Berg-Kirkpatrick, T., and Dubnov, S. Music SketchNet: Controllable music generation via factorized representations of pitch and rhythm. In *International Society for Music Information Retrieval (ISMIR)*, 2020.

Chen, K., Wu, Y., Liu, H., Nezhurina, M., Berg-Kirkpatrick, T., and Dubnov, S. MusicLDM: Enhancing novelty in text-to-music generation using beat-synchronous mixup strategies. *arXiv:2308.01546*, 2023.

Chen, T. On the importance of noise scheduling for diffusion models. Technical report, Google Research, 2023.

Chen, T., Xu, B., Zhang, C., and Guestrin, C. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

Choi, J., Choi, Y., Kim, Y., Kim, J., and Yoon, S. Custom-Edit: Text-guided image editing with customized diffusion models. *IEEE/CVF Conference on Computer Vision and Pattern Recognition - AI4CC Workshop*, 2023.

Chung, H., Kim, J., McCann, M. T., Klasky, M. L., and Ye, J. C. Diffusion posterior sampling for general noisy inverse problems. In *International Conference on Learning Representations (ICLR)*, 2023.

Clark, K., Vicol, P., Swersky, K., and Fleet, D. J. Directly fine-tuning diffusion models on differentiable rewards. *ArXiv*, abs/2309.17400, 2023.

Copet, J., Kreuk, F., Gat, I., Remez, T., Kant, D., Synnaeve, G., Adi, Y., and Défossez, A. Simple and controllable music generation. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Dai, S., Jin, Z., Gomes, C., and Dannenberg, R. Controllable deep melody generation via hierarchical music structure representation. In *International Society for Music Information Retrieval (ISMIR)*, 2021.

Defferrard, M., Benzi, K., Vandergheynst, P., and Bresson, X. FMA: A dataset for music analysis. In *International Society for Music Information Retrieval (ISMIR)*, 2017. URL `https://arxiv.org/abs/1612.01840`.

Dhariwal, P. and Nichol, A. Diffusion models beat GANs on image synthesis. *Neural Information Processing Systems (NeurIPS)*, 34, 2021.

Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., and Sutskever, I. Jukebox: A generative model for music. *arXiv:2005.00341*, 2020.

Dinh, L., Krueger, D., and Bengio, Y. NICE: Non-linear independent components estimation. *International Conference on Learning Representations (ICLR) Workshop*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. *International Conference on Learning Representations (ICLR)*, 2016.

Dong, H.-W., Hsiao, W.-Y., Yang, L.-C., and Yang, Y.-H. MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *AAAI Conference on Artificial Intelligence*, number 1, 2018.

Erickson, R. *Sound structure in music*. Univ of California Press, 1975.

Forsgren, S. and Martiros, H. Riffusion: Stable diffusion for real-time music generation, 2022. URL `https://riffusion.com/about`.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., and Cohen-Or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion, 2022.

Garcia, H. F., Seetharaman, P., Kumar, R., and Pardo, B. VampNet: Music generation via masked acoustic token modeling. In *International Society for Music Information Retrieval (ISMIR)*, 2023.

Gui, A., Gamper, H., Braun, S., and Emmanouilidou, D. Adapting frechet audio distance for generative music evaluation. *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023. URL `https://api.semanticscholar.org/CorpusID:265018955`.

Gupta, A., Yu, L., Sohn, K., Gu, X., Hahn, M., Li, F.-F., Essa, I., Jiang, L., and Lezama, J. Photorealistic video generation with diffusion models. 2023.

Hawthorne, C., Simon, I., Roberts, A., Zeghidour, N., Gardner, J., Manilow, E., and Engel, J. Multi-instrument music synthesis with spectrogram diffusion. In *International Society for Music Information Retrieval (ISMIR)*, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., and Cohen-Or, D. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. In *NeurIPS Workshop on Deep Gen. Models and Downstream Applications*, 2021.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Neural Information Processing Systems (NeurIPS)*, 33, 2020.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. *arXiv:2204.03458*, 2022.

Huang, Q., Park, D. S., Wang, T., Denk, T. I., Ly, A., Chen, N., Zhang, Z., Zhang, Z., Yu, J., Frank, C., et al. Noise2Music: Text-conditioned music generation with diffusion models. *arXiv:2302.03917*, 2023a.

Huang, R., Huang, J., Yang, D., Ren, Y., Liu, L., Li, M., Ye, Z., Liu, J., Yin, X., and Zhao, Z. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. *arXiv preprint arXiv:2301.12661*, 2023b.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022.

Karunratanakul, K., Preechakul, K., Aksan, E., Beeler, T., Suwajanakorn, S., and Tang, S. Optimizing diffusion noise can serve as universal motion priors. *arXiv preprint arXiv:2312.11994*, 2023.

Kawar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., and Irani, M. Imagic: Text-based real image editing with diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Kilgour, K., Zuluaga, M., Roblek, D., and Sharifi, M. Frechet audio distance: A metric for evaluating music enhancement algorithms. *arXiv:1812.08466*, 2018.

Kim, D., Lai, C.-H., Liao, W.-H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., and Ermon, S. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *ArXiv*, abs/2310.02279, 2023. URL https://api.semanticscholar.org/CorpusID:263622294.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2013.

Kumar, R., Seetharaman, P., Luebs, A., Kumar, I., and Kumar, K. High-fidelity audio compression with improved RVQGAN. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Lee, S.-g., Ping, W., Ginsburg, B., Catanzaro, B., and Yoon, S. Bigvgan: A universal neural vocoder with large-scale training. *arXiv preprint arXiv:2206.04658*, 2022.

Leemput, S. C. v., Teuwen, J., Ginneken, B. v., and Manniesing, R. Memcnn: A python/pytorch package for creating memory-efficient invertible neural networks. *Journal of Open Source Software*, 2019. ISSN 2475-9066. doi: 10.21105/joss.01576.

Levy, M., Giorgi, B. D., Weers, F., Katharopoulos, A., and Nickson, T. Controllable music production with diffusion models and guidance gradients. *ArXiv*, abs/2311.00613, 2023.

Liu, H., Chen, Z., Yuan, Y., Mei, X., Liu, X., Mandic, D., Wang, W., and Plumbley, M. D. AudioLDM: Text-to-audio generation with latent diffusion models. In *International Conference on Machine Learning (ICML)*, 2023a.

Liu, H., Tian, Q., Yuan, Y., Liu, X., Mei, X., Kong, Q., Wang, Y., Wang, W., Wang, Y., and Plumbley, M. D. AudioLDM 2: Learning holistic audio generation with self-supervised pretraining. *arXiv preprint arXiv:2308.05734*, 2023b.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *ArXiv*, abs/2211.01095, 2022.

Luo, S., Tan, Y., Huang, L., Li, J., and Zhao, H. Latent consistency models: Synthesizing high-resolution images with few-step inference. *ArXiv*, abs/2310.04378, 2023. URL https://api.semanticscholar.org/CorpusID:263831037.

McFee, B. and Ellis, D. Analyzing song structure with spectral clustering. In *International Society for Music Information Retrieval (ISMIR)*. Citeseer, 2014.

McFee, B., Barrington, L., and Lanckriet, G. R. Learning similarity from collaborative filters. In *International Society for Music Information Retrieval (ISMIR)*, 2010.

Mokady, R., Hertz, A., Aberman, K., Pritch, Y., and Cohen-Or, D. Null-text inversion for editing real images using guided diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Müller, M. *Fundamentals of music processing: Audio, analysis, algorithms, applications*. Springer, 2015.

Pan, Z., Gherardi, R., Xie, X., and Huang, S. Effective real image editing with accelerated iterative diffusion inversion. In *IEEE/CVF International Conference on Computer Vision (CVPR)*, 2023.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv*, 2022.

Prabhudesai, M., Goyal, A., Pathak, D., and Fragkiadaki, K. Aligning text-to-image diffusion models with reward backpropagation. *ArXiv*, abs/2310.03739, 2023.

Preechakul, K., Chatthee, N., Wizadwongsa, S., and Suwajanakorn, S. Diffusion autoencoders: Toward a meaningful and decodable representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Ronneberger, O., Fischer, P., and Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer Assisted Interventions (MICCAI)*, 2015.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Saharia, C., Chan, W., Chang, H., Lee, C., Ho, J., Salimans, T., Fleet, D., and Norouzi, M. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH Conference Proceedings*, 2022a.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. Photorealistic text-to-image diffusion models with deep language understanding. *Neural Information Processing Systems (NeurIPS)*, 35, 2022b.

Schneider, F., Jin, Z., and Schölkopf, B. Mo\^ usai: Text-to-music generation with long-context latent diffusion. *arXiv preprint arXiv:2301.11757*, 2023.

Si, C., Huang, Z., Jiang, Y., and Liu, Z. Freeu: Free lunch in diffusion u-net. *ArXiv*, abs/2309.11497, 2023.

Simonetta, F., Carnovalini, F., Orio, N., and Rodà, A. Symbolic music similarity through a graph-based representation. In *Audio Mostly 2018 on Sound in Immersion and Emotion*. 2018.

Skalse, J., Howe, N., Krasheninnikov, D., and Krueger, D. Defining and characterizing reward gaming. *Neural Information Processing Systems (NeuraIPS)*, 35, 2022.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2020.

Stevens, S. S., Volkmann, J., and Newman, E. B. A scale for the measurement of the psychological magnitude pitch. *Journal of the Acoustical Society of America (JASA)*, 1937.

Wallace, B., Gokul, A., Ermon, S., and Naik, N. V. End-to-end diffusion latent optimization improves classifier guidance. *IEEE/CVF International Conference on Computer Vision (ICCV)*, abs/2303.13703, 2023a.

Wallace, B., Gokul, A., and Naik, N. EDICT: Exact diffusion inversion via coupled transformations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023b.

Watson, D., Chan, W., Martin-Brualla, R., Ho, J., Tagliasacchi, A., and Norouzi, M. Novel view synthesis with diffusion models. *ArXiv*, abs/2210.04628, 2022.

Wu, S.-L., Donahue, C., Watanabe, S., and Bryan, N. J. Music controlnet: Multiple time-varying controls for music generation. *ArXiv*, abs/2311.07069, 2023a.

Wu, Y., Chen, K., Zhang, T., Hui, Y., Berg-Kirkpatrick, T., and Dubnov, S. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, 2023b.

Xia, W., Zhang, Y., Yang, Y., Xue, J.-H., Zhou, B., and Yang, M.-H. Gan inversion: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45, 2021.

Yu, J., Wang, Y., Zhao, C., Ghanem, B., and Zhang, J. Freedom: Training-free energy-guided conditional diffusion model. *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., and Tagliasacchi, M. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 30, 2021.

Zhang, L., Rao, A., and Agrawala, M. Adding conditional control to text-to-image diffusion models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Zhao, S., Chen, D., Chen, Y.-C., Bao, J., Hao, S., Yuan, L., and Wong, K.-Y. K. Uni-ControlNet: All-in-one control to text-to-image diffusion models. *arXiv:2305.16322*, 2023.

Zhu, G., Caceres, J.-P., Duan, Z., and Bryan, N. J. Musichifi: Fast high-fidelity stereo vocoding. *ArXiv*, abs/2403.10493, 2024. URL https://api.semanticscholar.org/CorpusID:268510221.

A. Diffusion Review

Denoising diffusion probabilistic models (DDPMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020) or diffusion models are a class of generative latent variable model. They are defined by a forward and reverse random Markov process. Intuitively, the forward process takes clean data and iteratively corrupts it with noise to train a (denoising) neural network and the reverse process takes random noise and iteratively refines it with the learned network to generate new data.

The forward process is defined as a Markov chain:

$$q(\boldsymbol{x}_0, ..., \boldsymbol{x}_T) := q(\boldsymbol{x}_0) \prod_{t=1}^{T} q(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}) \tag{4}$$

$$q(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}) := \mathcal{N}(\sqrt{1 - \beta_t} \boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \tag{5}$$

where $q(\boldsymbol{x}_0)$ is the true data distribution, $q(\boldsymbol{x}_T)$ is a standard normal Gaussian distribution, $0 < \ \beta_1 < \beta_2 < \cdots < \beta_T$ are noise schedule parameters, and T is the total number of noise steps. To improve the efficiency of the fixed forward data corruption process, (5) can be simplified to

$$q(\boldsymbol{x}_t | \boldsymbol{x}_0) := \mathcal{N}(\sqrt{\bar{\alpha}_t} \boldsymbol{x}_0, (1 - \bar{\alpha}_t) \boldsymbol{I}) \tag{6}$$

$$\boldsymbol{x}_t := \sqrt{\bar{\alpha}_t} \boldsymbol{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon} \,, \tag{7}$$

where $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_t$, and $\boldsymbol{\epsilon}$ is standard normal Gaussian noise, enabling forward sampling for any step t given clean data $\boldsymbol{x}_0$.

Given the forward process, we can specify a model distribution $p_\theta(\boldsymbol{x}_0)$ that approximates $q_\theta(\boldsymbol{x}_0)$. To make $p_\theta(\boldsymbol{x}_0)$ easy to sample from, we specify the data generation process to be a

$$p_\theta(\boldsymbol{x}_0) = \int p_\theta(\boldsymbol{x}_0, ..., \boldsymbol{x}_T) d\boldsymbol{x}_{1,...,T} \tag{8}$$

$$p_\theta(\boldsymbol{x}_0, ..., \boldsymbol{x}_T) := p_\theta(\boldsymbol{x}_T) \prod_{t=1}^{T} p_\theta^{(t)}(\boldsymbol{x}_{t-1} | \boldsymbol{x}) \tag{9}$$

where $\boldsymbol{x}_0, ..., \boldsymbol{x}_T$ are latent variables all in same data space.

Given the true data generation process (4) and model (9), we can train a neural network to recover the intermediate noisy data $\boldsymbol{x}_{t-1}$ given $\boldsymbol{x}_t$. More specifically, Ho et al. (Ho et al., 2020) showed that if we optimize the variational lower bound (Kingma & Welling, 2013) of our data likelihood and we reparameterize our problem to predict the noise $\boldsymbol{\epsilon}$, we can learn a suitable neural network $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ with parameters θ via minimizing the mean squared error via:

$$\mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}, t} \Big[\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \|_2^2 \Big] \,, \tag{10}$$

where t is the diffusion time-step.

Given a learned $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$, we can generate new data via the reverse diffusion process, a.k.a. sampling. To do so, we sample random Gaussian noise $\boldsymbol{x}_T \sim \mathcal{N}(0, I)$ and then iteratively refine it via

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \Big(\boldsymbol{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \Big) + \sigma_t \boldsymbol{\epsilon}, \tag{11}$$

until $t = 0$ to create our generated data $\boldsymbol{x}_0$ after T denoising iterations. To obtain high-quality generations, T is typically large (e.g., 1000), which results in a slow generation process.

To reduce the computational cost of sampling (inference), Song et al. (2020) proposed denoising diffusion implicit models (DDIM). DDIM uses an alternative variation optimization objective that itself yields an alternative sampling formulation

$$\begin{aligned} \boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{\boldsymbol{x}_t - \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \right) \\ + \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) + \sigma_t \boldsymbol{\epsilon}, \end{aligned} \tag{12}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, $\alpha_0 := 1$, and σ_t and different random noise scales. This formulation minimizes the number of sampling steps needed during inference (e.g., $50 \sim 100$) with minimal impact on generation quality. Furthermore, special cases of DDIM are then two fold 1) when $\sigma_t = \sqrt{(1-\alpha_{t-1})/(1-\alpha_t)}\sqrt{1-\alpha_t/\alpha_{t-1}}$, DDIM sampling refers back to basic DDPM sampling and 2) when $\sigma_t = 0$ the sampling process becomes fully deterministic.

To improve text conditioning, classifier-free guidance (CFG) can be used to blend conditional and unconditional generation outputs and trade-off conditioning strength, mode coverage, and sample quality (Ho & Salimans, 2021). When training a model with CFG, conditioning is randomly set to a null value a fraction of the time. During inference, the diffusion model output $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}_{\text{text}})$ is replaced with

$$\hat{\boldsymbol{\epsilon}}_{CFG} = w \cdot \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}_{\text{text}}) + (1-w) \cdot \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}_\emptyset), \tag{13}$$

where $\boldsymbol{c}_{\text{text}}$ are text embeddings, w is the CFG scaling factor, and $\boldsymbol{c}_\emptyset$ are null embeddings.

B. EDICT and DOODL with invertible layers

Exact Diffusion Inversion via Coupled Transformations, or EDICT, is a sampling method introduced in Wallace et al. (2023b) to enable *exact* diffusion inversion. EDICT accomplishes this by denoising two correlated diffusion chains, $\boldsymbol{x}'_t$ and $\boldsymbol{x}''_t$, at once, with the following updates:

$$\begin{aligned}
\boldsymbol{x}_t^{\prime\text{inter}} &= \sqrt{\frac{\alpha_{t-1}}{\alpha_t}}\boldsymbol{x}'_t + \left(\sqrt{1-\alpha_{t-1}} - \sqrt{\frac{\alpha_{t-1}(1-\alpha_t)}{\alpha_t}}\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}''_t, t) \\
\boldsymbol{x}_t^{\prime\prime\text{inter}} &= \sqrt{\frac{\alpha_{t-1}}{\alpha_t}}\boldsymbol{x}''_t + \left(\sqrt{1-\alpha_{t-1}} - \sqrt{\frac{\alpha_{t-1}(1-\alpha_t)}{\alpha_t}}\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^{\prime\text{inter}}, t) \\
\boldsymbol{x}'_{t-1} &= p\boldsymbol{x}_t^{\prime\text{inter}} + (1-p)\boldsymbol{x}_t^{\prime\prime\text{inter}} \\
\boldsymbol{x}''_{t-1} &= p\boldsymbol{x}_t^{\prime\prime\text{inter}} + (1-p)\boldsymbol{x}'_{t-1},
\end{aligned}$$

where the first two lines denote affine coupling layers and the last two lines are mixing layers with a fixed mixing coefficient p. This sampling procedure has the benefit of being exactly invertible:

$$\begin{aligned}
\boldsymbol{x}_{t+1}^{\prime\prime\text{inter}} &= \frac{\boldsymbol{x}''_t - (1-p)\boldsymbol{x}'_t}{p} \\
\boldsymbol{x}_{t+1}^{\prime\text{inter}} &= \frac{\boldsymbol{x}'_t - (1-p)\boldsymbol{x}_{t+1}^{\prime\prime\text{inter}}}{p} \\
\boldsymbol{x}''_{t+1} &= \sqrt{\frac{\alpha_{t+1}}{\alpha_t}}\left(\boldsymbol{x}_{t+1}^{\prime\prime\text{inter}} - \left(\sqrt{1-\alpha_t} - \sqrt{\frac{\alpha_t(1-\alpha_{t+1})}{\alpha_{t+1}}}\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{t+1}^{\prime\text{inter}}, t+1)\right) \\
\boldsymbol{x}'_{t+1} &= \sqrt{\frac{\alpha_{t+1}}{\alpha_t}}\left(\boldsymbol{x}_{t+1}^{\prime\text{inter}} - \left(\sqrt{1-\alpha_t} - \sqrt{\frac{\alpha_t(1-\alpha_{t+1})}{\alpha_{t+1}}}\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}''_{t+1}, t+1)\right)
\end{aligned}$$

One consequence of the dual-chain sampling approach is the inherent tradeoff in setting the p mixing parameter, as p needs to be sufficiently low to prevent the two chains from diverging (especially at low sampling steps), and sufficiently high to prevent numerical precision errors when inverting the chains.

In the official implementation for DOODL, EDICT's invertibility is not used, and instead normal checkpointing is used on the EDICT sampler, thus using 4x the number of model calls as standard backpropagation. However, given the invertible nature of EDICT, DOODL can alternatively be formulated to directly use the inverse operation rather than storing *all* function inputs in memory. In this setup, only the final $\boldsymbol{x}_0$ is stored in GPU memory, and then the inverse sampling operation is used to recalculate the function inputs, which are then passed back through the model to recalculate the intermediate activations for gradient calculation. This procedure is more memory efficient than the official implementation of DOODL and DITTO, yet *sextuples* the number of model calls and runtime, thus being the slowest procedure for inference-time latent optimization. Figure 5 describes both setups more in detail.

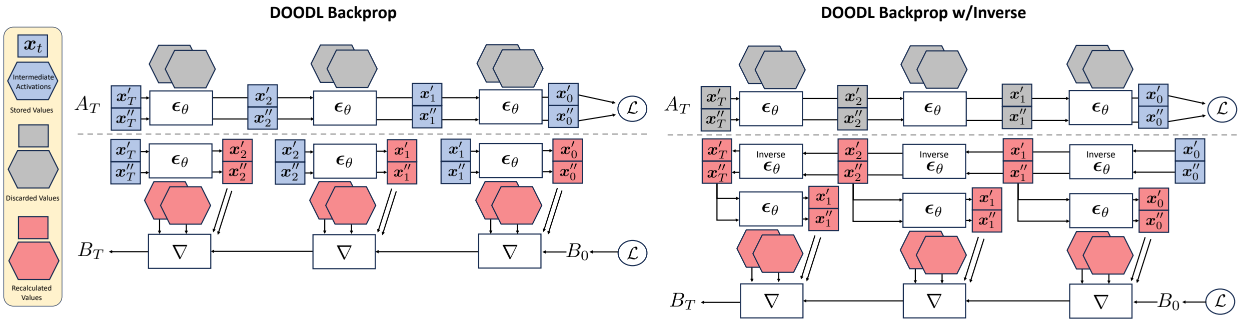

Figure 5. Forward and Backward pass for DOODL, both in its official implementation and alternatively by using the EDICT invertible layers. The standard DOODL backprop doubles the number of model calls (relative to DITTO) due to the EDICT sampling, yet uses checkpointing to store function inputs for each timestep. When utilizing EDICT's invertibility, only the final outputs are stored in memory, yet the inversion process requires two *more* model passes per timestep during the backwards pass.

C. Correlation-Based Intensity Control

Given the surprising poor control performance of Music ControlNet (Wu et al., 2023a) on the intensity control task despite being fully trained on such inputs, we investigated alternative metrics for understanding control adherence. Notably, we find that Music ControlNet implicitly models the intensity *correlation*, paying more attention to the overall shape of the intensity curve across time than the absolute dB values of the curve itself. We believe this makes sense, given the UNet backbone convolution (correlation) layers are both scale and location invariant. Given this result, we can alternatively parameterize intensity control to directly optimize for correlation by setting $\mathcal{L} \propto -\rho(f(\boldsymbol{x}_0), \boldsymbol{y})$, or by maximizing the correlation between the target and output intensity curves.

Table 6. Intensity correlation results for Music ControlNet and DITTO with both the standard and correlation-based loss function. By optimizing for correlation instead of absolute intensity, we can match the correlation of Music ControlNet while improving audio quality and text relevance.

Method	MSE (↓)	ρ (↑)	FAD (↓)	CLAP (↑)
Music ControlNet	38.4108	**0.9413**	11.1315	0.3084
DITTO ($\mathcal{L} \propto \lvert\lvert f(\boldsymbol{x}_0) - \boldsymbol{y}\rvert\rvert_2^2$)	**4.7576**	0.6166	**10.5294**	**0.4326**
DITTO ($\mathcal{L} \propto -\rho(f(\boldsymbol{x}_0), \boldsymbol{y})$)	60.8952	0.9040	11.0858	0.3503

In Table 6, we show both the absolute MSE and correlation ρ values for Music ControlNet, DITTO, and DITTO with the correlation based loss function. Music ControlNet has exceptional performance for intensity correlation, while baseline DITTO unsurprisingly prioritizes absolute intensity over correlation given its optimization objective. By switching to the correlation objective, DITTO can nearly match the correlation performance of Music ControlNet, all the while maintaining some of the absolute intensity DITTO's performance in audio quality and text relevance. This experiment shows how a single target feature can be parameterized in DITTO's flexible setup in multiple ways to change the intended behavior for rapid experimentation.

D. DITTO for Real-Audio Inversion

Inversion, or the task of encoding real reference media $\boldsymbol{x}_{\text{ref}}$ into a generative model's latent space, is crucial for image and audio editing tasks (Song et al., 2020; Dhariwal & Nichol, 2021; Xia et al., 2021; Mokady et al., 2023). Past audio-domain inversion work is very limited while past image-domain methods include naively adding noise to inputs (Song et al., 2020), reversing the DDIM sampling process (Dhariwal & Nichol, 2021), and learning additional *null-text* parameters to improve inversion accuracy (Mokady et al., 2023). We use DITTO for the task of inversion by setting $f(\boldsymbol{x}_0) = \boldsymbol{x}_0$, $\boldsymbol{y} = \boldsymbol{x}_{\text{ref}}$, and the loss to be the MSE or $\mathcal{L} \propto ||f(\boldsymbol{x}_0) - \boldsymbol{y}||_2^2$. Then, we can solve (2) to find an $\boldsymbol{x}_T$ such that (3) will produce $\boldsymbol{x}_0$ that reconstructs the target reference media $\boldsymbol{x}_{\text{ref}}$. While high-quality reconstruction is trivially possible with the fully invertible EDICT sampler (Wallace et al., 2023b), further editing with inverted content is complicated by its dual chain fully-deterministic sampling (Pan et al., 2023).

For generative text-conditioned models, a key factor of the inversion equation is the scale of the *classifier-free guidance* parameter, which helps improve controllability through text (Ho & Salimans, 2021), but noticeably makes the inversion process more difficult, as using classifier-free guidance results in diverging from the simple DDIM-based inversion (Mokady et al., 2023). Against DITTO, we compare with the Naïve inversion method of simply adding Gaussian noise to the reference spectrogram, the DDIM-based inversion which runs the DDIM sampling process in reverse through the model, and the recent Null-Text Inversion (Mokady et al., 2023) method, which starts with the DDIM inversion and then learns a time-dependent unconditional text embedding $\boldsymbol{c}_{\emptyset,t}$ to improve inversion results in the presence of high guidance scales. Like in null-text, we use the DDIM inversion as an initial guess for DITTO.

As the goal is direct recreation of the reference audio, we report MSE reconstruction across the entire 5K-sample MusicCaps dataset. We run this evaluation across four different guidance scales (ranging from 0, which is purely unconditional, to 7.5), and additionally run this on both our baseline 6 second model as well as a *24* second music generation model, which maintains all the same training hyperparameters and model size as our base model and only differs in that the output dimension is $2048 \times 160 \times 1$. In Table 7, we show that DITTO beats all other inversion methods across all guidance scales and model sizes, with the exception of the highest guidance scale on the 6 second base model, for which it performs slightly worse than null-text inversion. Notably, DITTO's superior performance on the 24 second model shows that scaling the number of free parameters with the image size (as $\boldsymbol{x}_T$ is the same shape as the output spectrogram) helps maintain reconstruction quality in the presence of high guidance, while methods that do not scale with the image size (like null-text inversion) do not have this benefit.

Qualitatively, we find that null-text inversion exhibits unique semantic artifacts in the reconstructed audio, such as replacing sung vocals with trumpets or tambourines with hi-hats, while DITTO avoids this failure case. As all the training data for the base model was on purely instrumental music, this shows that DITTO allows TTM diffusion models to interact with real audio outside the distribution of their training data. In further work, we hope to explore more complicated edits that require inverted inputs (which is common in the image domain) and thus compare against the EDICT-based approach.

Table 7. Inversion results across context size and guidance strength. DITTO performs SOTA reconstruction in most cases and noticeably scales with context size.

MSE (↓)	6 seconds				24 seconds			
	$w=0$	$w=1$	$w=4$	$w=7.5$	$w=0$	$w=1$	$w=4$	$w=7.5$
Naïve	0.0678	0.0668	0.0714	0.0787	0.1044	0.1042	0.1071	0.1122
DDIM	0.0115	0.0072	0.0192	0.0334	0.0089	0.0072	0.0115	0.0179
NT	0.0043	0.0072	0.0055	**0.0072**	0.0057	0.0072	0.0057	0.0060
DITTO (ours)	**0.0011**	**0.0010**	**0.0025**	0.0075	**0.0011**	**0.0011**	**0.0015**	**0.0023**

E. Reference-Free Looping

While we generally focus on long-form reference-based loop generation, where we seamlessly take existing audio and blend it back into itself, we note that DITTO can also be used for short-form reference-*free* loop generation, where we seek to generate a short musical loop unconditionally. This framework is similar to the reference-based looping, but instead defines the generated audio to loop back into *itself*, rather than into some fixed reference audio. More formally, we define $\mathbf{M}_{\text{gen},1}$ and $\mathbf{M}_{\text{gen},2}$ as two o sized masks over the generated spectrogram, and set $f(\boldsymbol{x}_0) = \mathbf{M}_{\text{gen},1} \odot \boldsymbol{x}_0$, $\boldsymbol{y} = \mathbf{M}_{\text{gen},2} \odot \boldsymbol{x}_0$, and $\mathcal{L} \propto \|f(\boldsymbol{x}_0) - \boldsymbol{y}\|_2^2$, such that the model optimizes to match the overlap region of its own generation during DITTO. We note that by setting $\mathbf{M}_{\text{gen},2}$ to occur earlier in the spectrogram (rather than one of the edges), we can generate loops of lengths that are less than or equal to the total context window (in our case, 6 seconds). In Figure 6, we show spectrograms of reference-free looping with an $o = 0.5$ second overlap and a total of two repetitions, with the loop boundary shown in red.

F. Musical Structure Transfer

While in the main paper, we focus our musical structure control task as controlling high-level musical form through simple musical phrase diagrams (like "ABA"), we can also directly *transfer* the structure of an existing song to our generation with

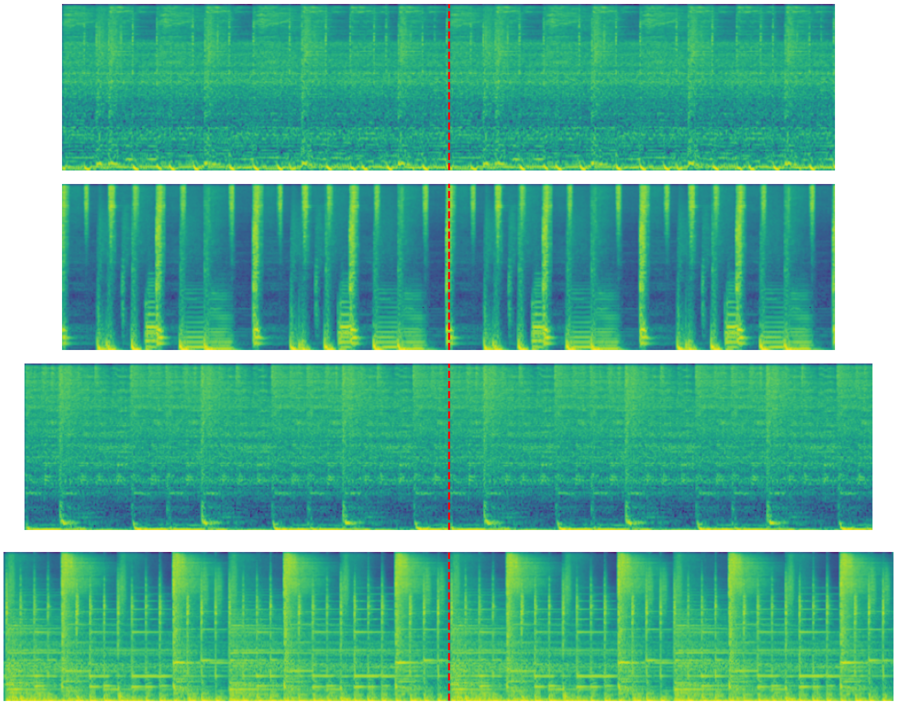

Figure 6. Reference-free loop generation with an overlap of $o = 0.5$ seconds. Loop boundary is shown in red.

DITTO through a similar process. Namely, instead of generating a target self-similarity matrix based on a given phrase diagram, we can instead set $\boldsymbol{y} = \mathbf{T}(y)\mathbf{T}(y)^\top$, where y is the mel-spectrogram of a *real* song and $\mathbf{T}(\cdot)$ is our MFCC-based timbre-extraction function. In this way, using $\mathcal{L} \propto \|f(\boldsymbol{x}_0) - \boldsymbol{y}\|_2^2$ we can use DITTO to generate music that matches the fine-grained self-similarity matrix of an existing musical fragment. Note that here we omit the 2D Savitzky-Golay step over the output self-similarity matrix, as here we want to directly match the intra-phrase similarity structures (rather than trying to capture broad musical form). We show examples of spectrograms with the target and generated self-similarity matrices in Fig. 7, where target self-similarity matrices are extracted from songs from the Free Music Archive dataset (Defferrard et al., 2017).

G. Alternative Sampling Methods

Unlike previous works on diffusion latent optimization (Wallace et al., 2023a), DITTO imposes no restrictions on the sampling process used to perform the optimization procedure, thus freeing us to choose any performant diffusion model sampling algorithm. Namely, we explore using DPM-Solver++ (Lu et al., 2022), a SOTA diffusion sampler for improving sample quality in conditional diffusion settings. Using outpainting and intensity control as test cases, in Table 8 we show MSE and FAD results. We interestingly find that DDIM is *better* than DPM++ for the intensity control task, yet DPM++ is slightly better for the outpainting task. We invite future work on discovering both theoretically and empirically how different diffusion sampling algorithms effect the noise latent optimization process.

Table 8. Comparison of different samplers for DITTO. DDIM works solidly better than DPM++ for the intensity task, and DPM++ preforms slightly better for outpainting.

Target	Sampler	MSE	FAD
Intensity	DDIM	4.77	10.53
Intensity	DPM++	6.30	11.04
Outpainting	DDIM	–	9.19
Outpainting	DPM++	–	9.12

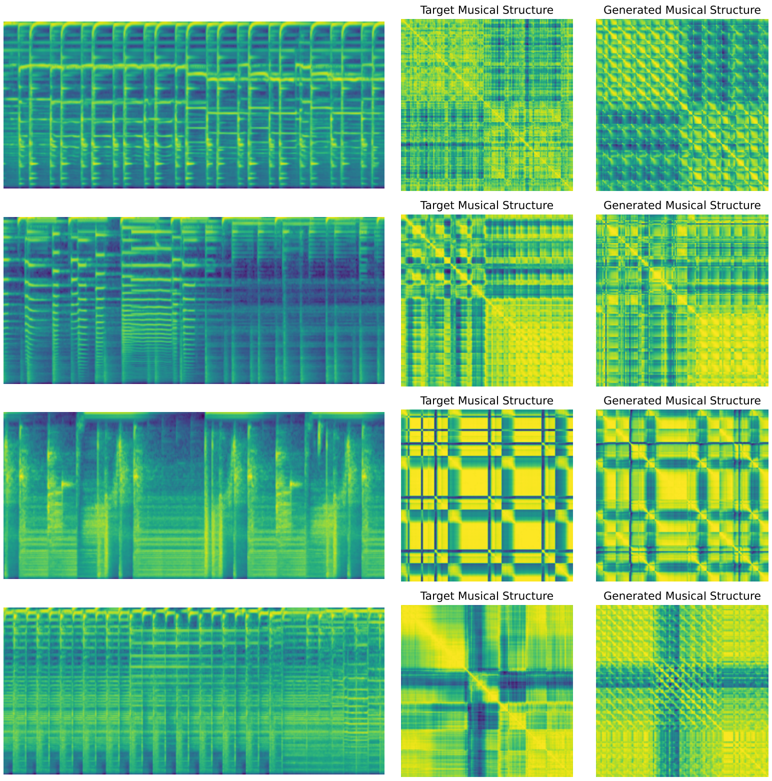

Figure 7. Musical Structure Transfer using self-similarity MFCC matrices extracted from real musical audio as the target.

H. Multi-Objective DITTO

Inspired by (Wu et al., 2023a), we can leverage the flexibility of DITTO to incorporate *multiple* feature matching criteria for a multi-objective optimization setup:

$$\boldsymbol{x}_T^* = \arg\min_{\boldsymbol{x}_T} \frac{1}{M} \sum_{i=1}^{M} \lambda_i \mathcal{L}_i \left(f_i(\boldsymbol{x}_0), \boldsymbol{y}_i \right), \tag{14}$$

where we include additional λ_i weights to balance the different scales of each loss function. Given DITTO's generality, this allows us to combine both editing *and* control signals at the same time, effectively unlocking the ability to iteratively compose long-form music with fine-grained temporal control. Here, we experiment with Intensity+Structure and Intensity+Melody, showing the combination of multiple reference-free controls, and Intensity+Outpainting, showing how reference-free controls can be composed with reference-based editing methods. For Intensity+Outpainting and Intensity+Structure we set $\lambda_{\text{intensity}} = 1/40$ and set $\lambda_{\text{intensity}} = 1/4$ for Intensity+Melody, while all other $\lambda_i = 1$, as intensity is calculated in the raw dB space. For the Intensity+Outpainting control, we use an overlap of $o = 2$ seconds and only optimize the intensity curve for the *nonoverlapping* section, having a similar effect to the “don't care” regions in Wu et al. (2023a). Here we compare against FreeDoM (Yu et al., 2023) for all tasks and Music-ControlNet (Wu et al., 2023a) for the Intensity+Melody task.

In Table 9, we find that FreeDoM in general struggles to follow multiple control signals across most tasks, while DITTO is able to more effectively balance the competing optimization objectives. Interestingly, we find generally low performance on the Intensity+Melody task across all methods, which leave for future work.

In Figures 8 and 9, we show spectrograms and output features for both experiments.

I. Reusing Optimized Latents

A key bottleneck of inference-time optimization methods like DITTO is the apparent need for the optimization procedure to generate a single output that matches the given feature, thus limiting its scalability. In order to mitigate this effect and

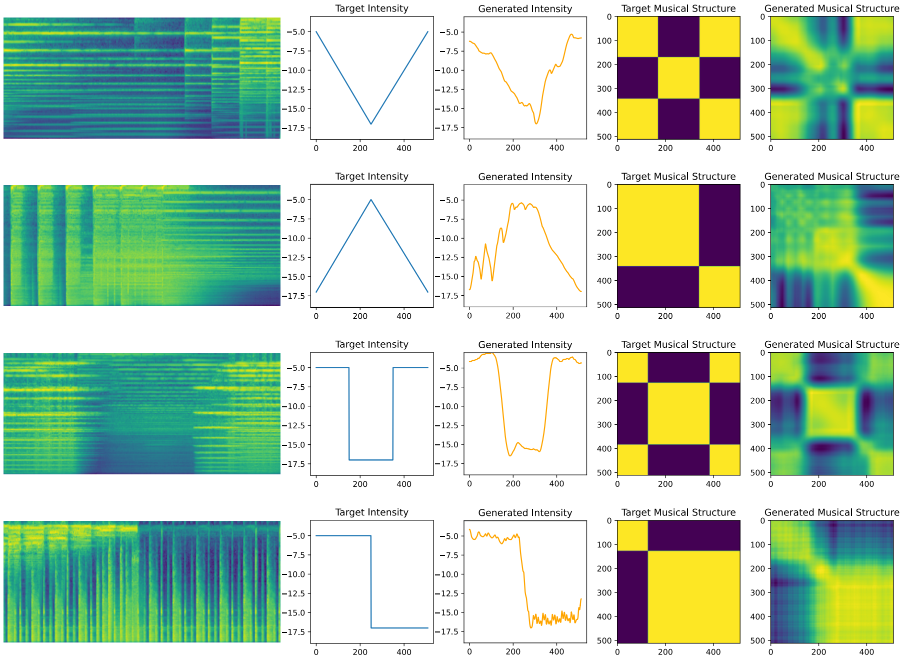

Figure 8. Output spectrograms, intensity curves, and MFCC self-similarity matrices for multi-objective DITTO with intensity and structure set as the feature extractors.

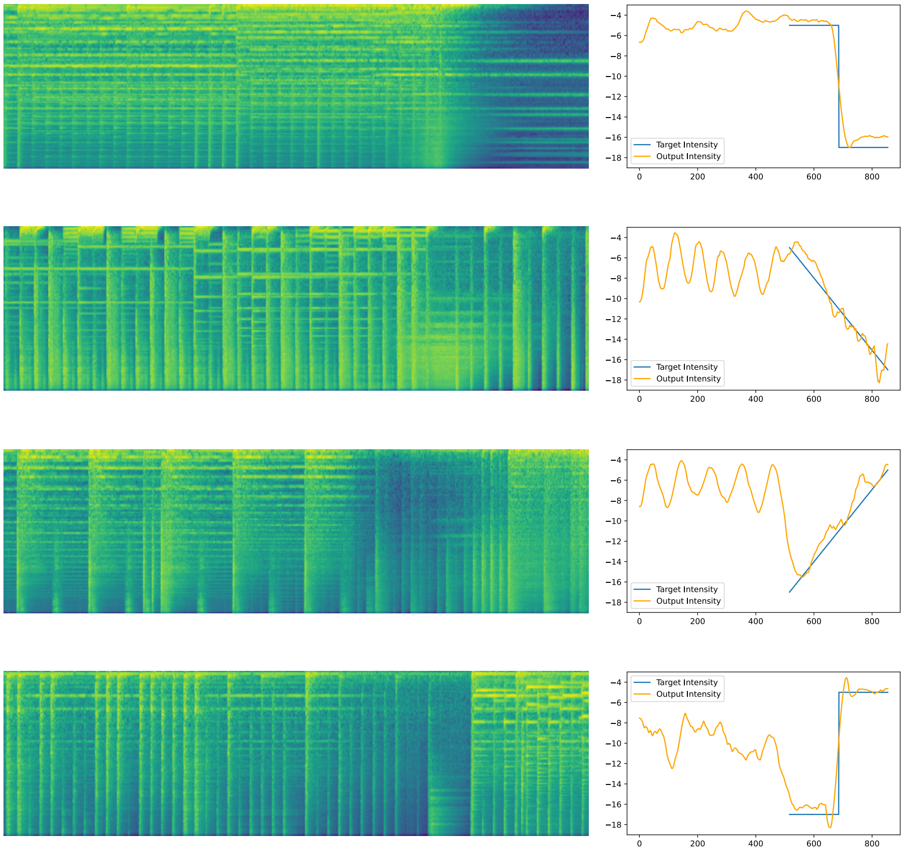

Figure 9. Output spectrograms and intensity curves for multi-objective DITTO with outpainting and intensity set as the feature extractors. The overlap is set to $o = 2$ seconds, and intensity control is only applied over the non-overlapping section.

Table 9. Multi-objective control results. DITTO More effectively balances multiple control signals than FreeDoM and Music ControlNet.

Control Method	**Intensity+Outpainting** Intensity MSE (↓)	FAD (↓)	CLAP (↑)
DITTO	**5.783**	**0.699**	**0.506**
FreeDoM	23.945	0.705	0.502
	Intensity+Structure Intensity MSE / Structure MSE (↓)	FAD (↓)	CLAP (↑)
DITTO	**6.802 / 0.092**	**0.661**	0.432
FreeDoM	21.033 / 0.304	0.669	**0.490**
	Intensity+Melody Intensity MSE (↓) / Melody Acc (↑)	FAD (↓)	CLAP (↑)
DITTO	**7.833** / 0.436	0.680	0.405
FreeDoM	21.185 / 0.198	**0.683**	**0.494**
Music-ControlNet	37.841 / **0.452**	0.604	0.347

accelerate the creative workflow for users, we explore how we can *reuse* optimized latents $\boldsymbol{x}_T^*$ to generate diverse outputs that follow the initial optimized feature signal.

A natural idea to add reusability to optimized latents is to treat each $\boldsymbol{x}_T^*$ as the mean of some normal distribution $\mathcal{N}(\boldsymbol{x}_T^*, \sigma^2)$ within the model's latent space for some hyperparameter σ^2, and then sample an $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T^*, \sigma^2)$ at inference time without re-optimizing. We find that this process leads to considerable divergence from the optimized feature in practice, and leave this to future work to explore further. Instead, we consider the case where we sample stochastic *trajectories* starting from $\boldsymbol{x}_T^*$, which in practice is as simple as switching to a stochastic sampling algorithm at inference time such as DDPM in (12) (note that we still use *deterministic* samplers during DITTO as stochastic samplers tend to make the optimization process considerably harder). Additionally, we also explore the case when the initial prompt $\boldsymbol{c}_{\text{text}}$ used during DITTO is varied, adding another source of stochasticity.

In this experiment, we compare two possible methods for reusing optimized latents for sampling stochastic trajectories: 1) after performing DITTO with DDIM, we sample using DDPM at inference time and 2) we use DDIM for optimization and DDPM for inference, but then additionally include the FreeDoM (Yu et al., 2023) guidance update in each DDPM step. To test reusability, after optimizing for each $\boldsymbol{x}_T^*$ given a target signal $\boldsymbol{y}$ and some text condition $\boldsymbol{c}_{\text{text}}$, we generate B samples $\boldsymbol{x}_0^{(i)}$ using $\boldsymbol{x}_T^*$ as the starting latent and our stochastic sampling algorithm of choice, and measure $\frac{1}{B}\sum_{i=1}^{B}\mathcal{L}(f(\boldsymbol{x}_0^{(i)}, \boldsymbol{y}))$, or the average loss over the stochastic samples, where *no* optimization is occuring. We perform this experiment both where each $\boldsymbol{x}_0^{(i)}$ is generated with a random prompt $\boldsymbol{c}_i$, and when each prompt is fixed to the initial prompt $\boldsymbol{c}_i = \boldsymbol{c}_{\text{text}}$ to measure the effect of additional stochasticity from conditioning.

In Table 10, we show results for intensity, melody, and musical structure control with a batch size $B = 10$. Notably, while switching to baseline DDPM during sampling predictably worsens the feature adherence, using FreeDoM with DDPM and starting at $\boldsymbol{x}_T^*$ yields significantly improved feature adherence to the optimized target. This presents a useful marriage of guidance-based and optimization-based approaches, as DITTO latents can act as reasonable feature priors by utilizing FreeDoM to guide the trajectory from the strong starting point.

J. Diffusion Latents and Low-Frequency Content

In Si et al. (2023), the authors discover that much of the low-frequency (in the 2D pixel domain) content of TTI model generations are determined exceedingly early on in the sampling process, where further sampling steps only produce high-frequency information and improve quality. This presents a compelling case for why DITTO has such strong expressivity: because many target controls for TTM generation like intensity, melody and musical structure are low-frequency features in the spectrogram domain (i.e. most high-frequency 2D content in spectrograms address audio quality factors), optimizing $\boldsymbol{x}_T$ to target these features is well within the diffusion model's latent space which already encodes low-frequency information in the first place. This is compounded by the fact that music tags and captions generally only address high-level stylistic

Table 10. Loss on samples generated with stochastic sampling from $\boldsymbol{x}_T^*$. We observe that DITTO latents natively can act as generalized feature priors, using FreeDoM on optimized latents to significantly improve feature adherence, thus showing how optimization-based and guidance-based methods can be used in conjunction for high-quality and efficient control.

Optimization `Sampler`	Inference `Sampler`	Feature	$\mathcal{L}$	$\mathcal{L}$ (Fixed Prompt)
DDIM	DDPM	Intensity	24.5120	13.8316
DDIM	DDPM+FreeDoM	Intensity	16.9780	11.2481
DDIM	DDPM	Melody	2.7973	2.7441
DDIM	DDPM+FreeDoM	Melody	1.8482	1.8710
DDIM	DDPM	Musical Structure	0.2952	0.2643
DDIM	DDPM+FreeDoM	Musical Structure	0.0251	0.0235

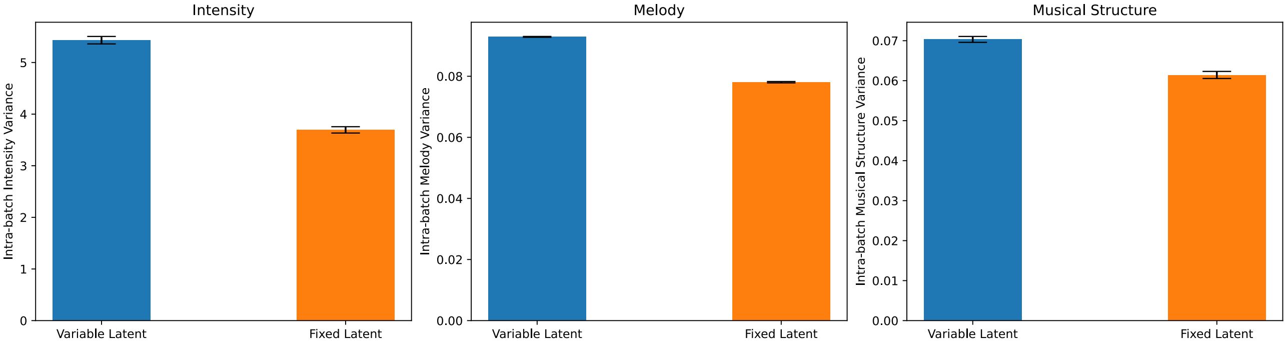

Figure 10. Intra-batch variance for model generations both with and without fixing the initial latent. We find a statistically significant effect that fixing the latent reduces feature variance, showing that $\boldsymbol{x}_T$ already encodes a great deal of feature information.

information, leaving everything that is not captured by the text captions (such as time-varying intensity, melody, and structure) to be incorporated into the initialization.

To validate this proposed justification, we generate 5K batches ($B = 10$) of samples from our base diffusion model, where half of the batches (2.5K) have random initializations and random prompts while the other half have the same initialization $\boldsymbol{x}_T$ (and still random prompts). For each group, we measure variance within each batch of the intensity, melody, and musical structure features extracted from the batch outputs. Shown in Fig. 10, we find a statistically significant effect across all features that fixing the initialization significantly reduces the intra-batch feature variance. This serves as empirical justification that to a certain extent, the model output's salient musical features are already determined *at initialization*.

K. Model Pre-training

For our spectrogram generation model, we follow an identical training processed to default TTM as to Music ControlNet (Wu et al., 2023a). We use a convolutional UNet (Ronneberger et al., 2015) with 5 2D-convolution ResNet (He et al., 2016) blocks with $[64, 64, 128, 128, 256]$ feature channels per block with a stride of 2 in between downsampling blocks. The UNet inputs Mel-scaled (Stevens et al., 1937) spectrograms clipped to a dynamic range of 160 dB and scaled to $[-1, 1]$ computed from 22.05 kHz audio with a hop size of 256 (i.e., frame rate $\mathrm{f_k} \approx 86$ Hz), a window size of 2048, and 160 Mel bins. For our genre, mood, and tempo global style control $\boldsymbol{c}_{\text{text}}$, we use learnable class-conditional embeddings with dimension of 256 that are injected into the inner two ResNet blocks of the U-Net via cross-attention. We use a cosine noise schedule with 1000 diffusion steps that are injected via sinusoidal embeddings with a learnable linear transformation summed directly with U-Net features in each block. We set our output time dimension to 512 or ≈6 seconds, yielding a 512×160×1 output dimension. We use an L1 training objective between predicted and actual added noise, an Adam optimizer with learning rate to 10^{-5} with linear warm-up and cosine decay. Due to limited data and efficiency considerations, we instantiate a relatively small model of 41M parameters and pre-train with distributed data parallel for 5 days on 32 A100 GPUs with a batch size of 24 per GPU. Finally, we also use MusicHifi (Zhu et al., 2024) as the vocoder: MusicHifi uses a BigVGAN vocoder (Lee et al., 2022) modified with a DAC discriminator (Kumar et al., 2023), trained with an AdamW optimizer with learning rate 0.0001, exponential learning rate decay on both our discriminator and generator optimizer, batch size of 48 per GPU, and 1536 channels for the initial upsampling layer that was trained on 8 A100 GPUs for 5 days.

L. Efficiency Experiment Details and Discussion

We run the test on a single 40GB A100 with $K = 70$ maximum optimization steps and $\tau = 2$ dB. For DOODL, we use a mixing coefficient of $p = 0.93$ at 50 steps following Wallace et al. (2023a) and $p = 0.83$ at 20 steps due to severe divergence issues with higher p at 20 steps.

Table 11. Speed comparison of various training-based, guidance-based, and optimization-based methods on the intensity control task, both in fine-tuning cost (in 40GB A100 GPU hours) and latency.

Method	Fine-tuning Cost (GPU Hours, ↓)	Latency (seconds, ↓)
Base TTM	-	0.612
ControlNet (Wu et al., 2023a)	576	1.456
FreeDoM (Yu et al., 2023)	0	2.867
DITTO (Ours)	0	82.192
DOODL (Wallace et al., 2023a)	0	206.897

To augment the analysis in Sec. L and display how DITTO's speed compares to other control methods for TTM diffusion models, in Table 11, we report both the latency (i.e. the time for a single sample to be generated, in seconds) and the overall fine-tuning cost in 40GB NVIDIA A100 GPU hours for our Base TTM model as well as DITTO, DOODL, Music ControlNet, and FreeDoM. Notably, Music ControlNet presents the lowest inference latency at the cost of over 500 GPU hours of fine-tuning. Of the training-free methods, DITTO is faster than DOODL but still ≈30x slower than FreeDoM, offering a clear trade-off in terms of latency and control strength.